

# Parallel Bayesian Global Optimization of Expensive Functions\*

Jialei Wang<sup>†1</sup>, Scott C. Clark<sup>‡2</sup>, Eric Liu<sup>§3</sup>, and Peter I. Frazier<sup>¶1</sup>

<sup>1</sup>School of Operations Research and Information Engineering, Cornell University

<sup>2</sup>SigOpt, 244 Kearny St, San Francisco, CA

<sup>3</sup>Yelp, Inc., 140 New Montgomery, San Francisco, CA

## Abstract

We consider parallel global optimization of derivative-free expensive-to-evaluate functions, and propose an efficient method based on stochastic approximation for implementing a conceptual Bayesian optimization algorithm proposed by Ginsbourger et al. (2007). To accomplish this, we use infinitesimal perturbation analysis (IPA) to construct a stochastic gradient estimator and show that this estimator is unbiased. We also show that the stochastic gradient ascent algorithm using the constructed gradient estimator converges to a stationary point of the  $q$ -EI surface, and therefore, as the number of multiple starts of the gradient ascent algorithm and the number of steps for each start grow large, the one-step Bayes optimal set of points is recovered. We show in numerical experiments that our method for maximizing the  $q$ -EI is faster than methods based on closed-form evaluation using high-dimensional integration, when considering many parallel function evaluations, and is comparable in speed when considering few. We also show that the resulting one-step Bayes optimal algorithm for parallel global optimization finds high quality solutions with fewer evaluations than a heuristic based on approximately maximizing the  $q$ -EI. A high quality open source implementation of this algorithm is available in the open source Metrics Optimization Engine (MOE).

## 1 Introduction

We consider derivative-free global optimization of expensive functions, in which (1) our objective function is time-consuming to evaluate, limiting the number of function evaluations we can perform;

---

\*Peter Frazier and Jialei Wang were partially supported by NSF CAREER CMMI-1254298, NSF CMMI-1536895, NSF IIS-1247696, AFOSR FA9550-12-1-0200, AFOSR FA9550-15-1-0038, and AFOSR FA9550-16-1-0046.

<sup>†</sup>jw865@cornell.edu

<sup>‡</sup>scott@sigopt.com

<sup>§</sup>eliu@yelp.com

<sup>¶</sup>pf98@cornell.edu

(2) evaluating the objective function provides only the value of the objective, and not the gradient or Hessian; (3) we seek a global, rather than a local, optimum. Such problems typically arise when the objective function is evaluated by running a complex computer code (see, e.g., Sacks et al. (1989)), but also arises when the objective function can only be evaluated by performing a laboratory experiment, or building a prototype system to be evaluated in the real world. In this paper we assume our function evaluations are deterministic, i.e., free from noise.

Bayesian Global Optimization (BGO) methods constitute one class of methods attempting to solve such problems. These methods were initially proposed by Kushner (1964), with early work being pursued in Mockus et al. (1978) and Mockus (1989), and more recent work including improved algorithms in Boender and Kan (1987), Jones et al. (1998) and Huang et al. (2006), convergence analysis in Calvin (1997), Calvin and Žilinskas (2002) and Vazquez and Bect (2010), and allowing noisy function evaluations in Calvin et al. (2005), Villemonteix et al. (2009), Frazier et al. (2009) and Huang et al. (2006).

The most well-known BGO method is Efficient Global Optimization (EGO) from Jones et al. (1998), which chooses each point at which to evaluate the expensive objective function in the “outer” expensive global optimization problem by solving an “inner” optimization problem: maximize the “expected improvement”. Expected improvement is the value of information Howard (1966) from a single function evaluation, and quantifies the benefit that this evaluation provides in terms of revealing a point with a better objective function value than previously known. If this is the final point that will be evaluated in the outer optimization problem, and if additional conditions are satisfied (the evaluations are free from noise, and the implementation decision, i.e., the solution that will be implemented in practice after the optimization is complete, is restricted to be a previously evaluated point), then the point with largest expected improvement is the Bayes-optimal point to evaluate, in the sense of providing the best possible average-case performance in the outer expensive global optimization problem Frazier and Wang (2016).

Solving EGO’s inner optimization problem is facilitated by an easy-to-compute and easy-to-differentiate expression for the expected improvement in terms of the (scalar) normal cumulative distribution function. Fast evaluation of the expected improvement and its gradient make it possible in many applications to solve the inner optimization problem in significantly less time than required per evaluation of the expensive outer objective, which is critical to EGO’s usefulness as an optimization algorithm.

The inner optimization problem at the heart of EGO, and its objective, the expected improvement, was generalized by Ginsbourger et al. (2007) to the parallel setting, in which the expensive objective can be evaluated at several points simultaneously. This generalization, called the “multi-points expected improvement” or the  $q$ -EI, is consistent with the decision-theoretic derivation of expected improvement, and quantifies the expected utility that will result from the evaluation of a *set* of points. Ginsbourger et al. (2007) also provided an analytical formula for the case when  $q = 2$ , or  $2$ -EI.

If the generalized inner optimization problem proposed by Ginsbourger et al. (2007), which is to find the set of points to evaluate next that jointly maximize the  $q$ -EI, could be solved efficiently, then this would provide the one-step Bayes-optimal set of points to evaluate in the outer problem, and would create a one-step Bayes-optimal algorithm for global optimization of expensive functions able to fully utilize parallelism.

This generalized inner optimization problem is challenging, however, because unlike the (scalar) expected improvement used by EGO, the  $q$ -EI lacks an easy-to-compute expression, and is calculable only through Monte Carlo simulation, high-dimensional numerical integration, or expressions involving high-dimensional multivariate normal cumulative distribution functions. This significantly restricts the set of applications in which a naive implementation can solve the inner problem faster

than a single evaluation of the outer optimization problem. Stymied by this difficulty, Ginsbourger et al. (2007), as well as the later work in Chevalier and Ginsbourger (2013), propose heuristic methods that are *motivated* by the one-step optimal algorithm of evaluating the set of points that jointly maximize the  $q$ -EI, but that do not actually achieve this gold standard.

**Contributions** The main contribution of this work is to provide a method that solves the inner optimization problem of maximizing the  $q$ -EI efficiently, creating a practical and broadly applicable one-step Bayes-optimal algorithm for parallel global optimization of expensive functions. To accomplish this we use infinitesimal perturbation analysis (IPA) (see Ho (1987)) to construct a stochastic gradient estimator of the gradient of the  $q$ -EI surface, and show that this estimator is unbiased, with a bounded second moment. Our method uses this estimator within a stochastic gradient ascent algorithm, and we show that it converges to a stationary point of the  $q$ -EI surface. We use multiple restarts to identify multiple stationary points, and then use ranking and selection to identify the best stationary point found. As the number of restarts and the number of iterations of stochastic gradient ascent within each start both grow large, the one-step optimal set of points to evaluate is recovered.

Our method can be implemented in both synchronous environments, in which function evaluations are performed in batches and finish at the same time, and asynchronous ones, in which a function evaluation may finish before others are done.

In addition to our methodological contribution, we have developed a high-quality open source software package, the “Metrics Optimization Engine (MOE)” Clark et al. (2014), implementing our method for solving the inner optimization problem, and the resulting algorithm for parallel global optimization of expensive functions. To further enhance computational speed, the implementation takes advantage of parallel computing, and achieves 100X speedup over single-threaded computation when deployed on a graphical processing unit (GPU). This software package has been used by Yelp and Netflix to solve global optimization problems arising in their businesses Clark (2014), Amatriain (2014). For the rest of the paper, we refer to our method as “MOE-qEI” because it is implemented in MOE.

We compare MOE-qEI against several benchmark methods. We show that MOE-qEI provides high-quality solutions to the outer optimization problem more quickly than the heuristic CL-mix policy proposed by Chevalier and Ginsbourger (2013), which is motivated by the inner optimization problem. We also show that MOE-qEI provides a substantial parallel speedup over the single-threaded EGO algorithm, which is one-step optimal when parallel resources are unavailable. We also compare our simulation-based method for solving the inner optimization problem against methods based on exact evaluation of the  $q$ -EI from Chevalier and Ginsbourger (2013) and Marmin et al. (2015) (discussed in more detail below) and show that our simulation-based approach to solving the inner optimization problem provides solutions to both the inner and outer optimization problem that are comparable in quality and speed when  $q$  is small, and superior when  $q$  is large.

**Related Work** Developed independently and in parallel with our work is Chevalier and Ginsbourger (2013), which provides a closed-form formula for computing  $q$ -EI, and the book chapter Marmin et al. (2015), which provides a closed-form expression for its gradient. Both require multiple calls to high-dimensional multivariate normal cumulative distribution functions (cdfs). These expressions can be used within an existing continuous optimization algorithm to solve the inner optimization problem that we consider.

While attractive in that they provide closed-form expressions, calculating these expressions when  $q$  is even moderately large is slow and numerically challenging. This is because calculating the

multivariate normal cdf in moderately large dimension is itself challenging, with state of the art methods relying on numerical integration or Monte Carlo sampling as described in Genz (1992). Indeed, the method for evaluating the  $q$ -EI from Chevalier and Ginsbourger (2013) requires  $q^2$  evaluations of the  $q - 1$  dimensional multivariate normal cdf, and the method for evaluating its gradient requires  $O(q^4)$  calls to multivariate normal cdfs with dimension ranging from  $q - 3$  to  $q$ . In our numerical experiments, we demonstrate that our method for solving the inner optimization problem requires less computation time and parallelizes more easily than do these competing methods, for  $q > 4$ , and performs comparably when  $q$  is smaller. We also demonstrate that MOE-qEI’s improved performance in the inner optimization problem for  $q > 4$  translates to improved performance in the outer optimization problem.

Other related work includes the previously proposed heuristic CL-mix from Chevalier and Ginsbourger (2013), which does not solve the inner maximization of  $q$ -EI, instead using an approximation. While solving the inner maximization of  $q$ -EI as we do makes it more expensive to compute the set of points to evaluate next, we show in our numerical experiments that it results in a substantial savings in the number of evaluations required to find a point with a desired quality. When function evaluations are expensive, this results in a substantial reduction in overall time to reach an approximately optimal solution.

In other related work on parallel Bayesian optimization, Frazier et al. (2011) and Xie et al. (2013) proposed a Bayesian optimization algorithm that evaluates pairs of points in parallel, and is one-step Bayes-optimal in the noisy setting under the assumption that one can only observe noisy function values for single points, or noisy function value differences between pairs of points. This algorithm, however, is limited to evaluating pairs of points, and does not extend to a higher level of parallelism.

There are also other non-Bayesian algorithms for derivative-free global optimization of expensive functions with parallel function evaluations from Dennis and Torczon (1991), Kennedy (2010) and Holland (1992). These are quite different in spirit from the algorithm we develop, not being derived from a decision-theoretic foundation.

**Outline** We begin in Section 2 by precisely describing the mathematical setting in which Bayesian global optimization is performed, and then defining the  $q$ -EI and the one-step optimal algorithm. We construct our stochastic gradient in Section 3.2, and combine this estimator together with stochastic gradient ascent to define a one-step optimal method for parallel Bayesian global optimization in Section 3.3. Then in Section 4.1 we show that the constructed gradient estimator of the  $q$ -EI surface is unbiased under certain mild regularity conditions. Moreover, in Section 4.2 we provide convergence analysis of the stochastic gradient ascent algorithm. Finally, in Section 5 we present numerical experiments: we compare MOE-qEI against previously proposed heuristics from the literature; we demonstrate that MOE-qEI provides a speedup over single-threaded EGO; we show that MOE-qEI is more efficient than optimizing evaluations of the  $q$ -EI using closed-form formula provided in Chevalier and Ginsbourger (2013) when  $q$  is large; and we show that MOE-qEI computes the gradient of  $q$ -EI faster than evaluating the closed-form expression proposed in Marmin et al. (2015).

## 2 Problem formulation and background

In this section, we describe a decision-theoretic approach to Bayesian global optimization in parallel computing environments, previously proposed by Ginsbourger et al. (2007). This approach was considered to be purely conceptual in Ginsbourger et al. (2007), as it contains a difficult-to-solve optimization sub-problem (our so-called “inner” optimization problem). In this section, we present

this inner optimization problem as background, and present a novel method in the subsequent section that solves it efficiently.

## 2.1 Bayesian Global Optimization

In Bayesian global optimization, one considers optimization of a function  $f$  with domain  $\mathbb{A} \subseteq \mathbb{R}^d$ . Our overarching goal is to find an approximate solution to

$$\min_{\mathbf{x} \in \mathbb{A}} f(\mathbf{x}).$$

We suppose that evaluating  $f$  is expensive or time-consuming, and that these evaluations provide only the value of  $f$  at the evaluated point, and not its gradient or Hessian. We assume that the function defining the domain  $\mathbb{A}$  is easy-to-evaluate, and that projections from  $\mathbb{R}^d$  into the nearest point in  $\mathbb{A}$  can be performed quickly.

Rather than focusing on asymptotic performance as the number of function evaluations grows large, we wish to find an algorithm that performs well, on average, given a limited budget of function evaluations. To formalize this, we model our prior beliefs on the function  $f$  with a Bayesian prior distribution, and we suppose that  $f$  was drawn at random by nature from this prior distribution, before any evaluations were performed. We then seek to develop an optimization algorithm that will perform well, on average, when applied to a function drawn at random from this prior.

## 2.2 Gaussian process priors

For our Bayesian prior distribution on  $f$ , we adopt a Gaussian process prior (see Rasmussen and Williams (2006)), which is specified by its mean function  $\mu(\mathbf{x}) : \mathbb{A} \rightarrow \mathbb{R}$  and positive semi-definite covariance function  $k(\mathbf{x}, \mathbf{x}') : \mathbb{A} \times \mathbb{A} \rightarrow \mathbb{R}$ . We write the Gaussian process as

$$f \sim \mathcal{GP}(\mu, k).$$

Then for a collection of points  $\mathbf{X} := (\mathbf{x}_1, \dots, \mathbf{x}_q)$ , the prior of  $f$  at  $\mathbf{X}$  is

$$f(\mathbf{X}) \sim \mathcal{N}(\boldsymbol{\mu}^{(0)}, \boldsymbol{\Sigma}^{(0)}), \tag{1}$$

where  $\boldsymbol{\mu}_i^{(0)} = \mu(\mathbf{x}_i)$  and  $\boldsymbol{\Sigma}_{ij}^{(0)} = k(\mathbf{x}_i, \mathbf{x}_j)$ ,  $i, j \in \{1, \dots, q\}$ .

Our proposed method for choosing the points to evaluate next additionally requires that  $\mu$  and  $k$  satisfy some mild regularity assumptions discussed below, but otherwise adds no additional requirements on  $\mu$  and  $k$ . When Bayesian global optimization is used in practice,  $\mu$  and  $k$  are typically chosen using an empirical Bayes approach discussed in Brochu et al. (2010), in which first, a parametrized functional form for  $m$  and  $k$  is assumed; second, a first stage of data is collected in which  $f$  is evaluated at points chosen according to a Latin hypercube or uniform design; and third, maximum likelihood estimates for the parameters specifying  $m$  and  $k$  are obtained. In some implementations, these estimates are updated iteratively as more evaluations of  $f$  are obtained. We adopt this method in our numerical experiments below in section 5, and describe it in more detail there. However, the specific contribution of this paper, a new method for solving an optimization sub-problem arising in the choice of design points, works with any choice of mean function  $\mu$  and covariance matrix  $k$ , as long as they satisfy the mild regularity conditions discussed below.

In addition to the prior distribution specified in (1), we may also have some previously observed function values, say  $y^{(i)} = f(\mathbf{x}^{(i)})$ , for  $i = 1, \dots, n$ . These might have been obtained through the previously mentioned first stage of sampling, or running the second stage sampling method we are

about to describe, or from some additional runs of the expensive objective function  $f$  performed by another party outside of the control of our algorithm. If no additional function values are available, we set  $n = 0$ . We define notation  $\mathbf{x}^{(1:n)} = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)})$  and  $y^{(1:n)} = (y^{(1)}, \dots, y^{(n)})$ .

We then combine these previously observed function values with our prior to obtain a posterior distribution on  $f(\mathbf{X})$ . This posterior distribution is still a multivariate normal (e.g., see Eq. (A.6) on pp. 200 in Rasmussen and Williams (2006))

$$f(\mathbf{X}) \mid \mathbf{X}, \mathbf{x}^{(1:n)}, y^{(1:n)} \sim \mathcal{N}(\boldsymbol{\mu}^{(n)}, \boldsymbol{\Sigma}^{(n)}), \quad (2)$$

with

$$\begin{aligned} \boldsymbol{\mu}^{(n)} &= \boldsymbol{\mu}^{(0)} + K(\mathbf{X}, \mathbf{x}^{(1:n)}) K(\mathbf{x}^{(1:n)}, \mathbf{x}^{(1:n)})^{-1} (y^{(1:n)} - \boldsymbol{\mu}^{(1:n)}), \\ \boldsymbol{\Sigma}^{(n)} &= K(\mathbf{X}, \mathbf{X}) - K(\mathbf{X}, \mathbf{x}^{(1:n)}) K(\mathbf{x}^{(1:n)}, \mathbf{x}^{(1:n)})^{-1} K(\mathbf{x}^{(1:n)}, \mathbf{X}), \end{aligned} \quad (3)$$

where  $\boldsymbol{\mu}^{(1:n)}$  are prior means on  $\mathbf{x}^{(1:n)}$ ,  $K(\mathbf{X}, \mathbf{x}^{(1:n)})$  is a  $q \times n$  matrix with  $K(\mathbf{X}, \mathbf{x}^{(1:n)})_{ij} = k(\mathbf{x}_i, \mathbf{x}^{(j)})$ , and similar for  $K(\mathbf{x}^{(1:n)}, \mathbf{X})$ ,  $K(\mathbf{X}, \mathbf{X})$  and  $K(\mathbf{x}^{(1:n)}, \mathbf{x}^{(1:n)})$ .

### 2.3 Multi-points expected improvement ( $q$ -EI)

In a parallel computing environment, we wish to use this posterior distribution to choose the set of points to evaluate next. Ginsbourger et al. (2007) proposed making this choice using a decision-theoretic approach, in which we consider the utility that evaluating a particular candidate set of points would provide, in terms of their ability to reveal points with objective function values better than previously known. We review this decision-theoretic approach here, and then present a new algorithm for implementing this choice in the next section.

Let  $q$  be the number of function evaluations that we may perform in parallel, and let  $\mathbf{X}$  be a candidate set of points that we are considering evaluating next. Let  $f_n^* = \min_{m \leq n} f(\mathbf{x}^{(m)})$  indicate the value of the best point evaluated, before beginning these  $q$  new function evaluations. The value of the best point evaluated after all  $q$  function evaluations are complete will be  $\min(f_n^*, \min_{i=1, \dots, q} f(\mathbf{x}_i))$ . The difference between these two values (the values of the best point evaluated, before and after these  $q$  new function evaluations) is called the *improvement*, and is equal to  $(f_n^* - \min_{i=1, \dots, q} f(\mathbf{x}_i))^+$ , where  $a^+ = \max(a, 0)$  for  $a \in \mathbb{R}$ .

We then value a joint set of evaluations at these candidate points  $\mathbf{X}$  as the expected value of this improvement, and we refer to this quantity as the *multi-points expected improvement* or  $q$ -EI from Ginsbourger et al. (2007). This multi-points expected improvement can be written as,

$$q\text{-EI}(\mathbf{X}) = \mathbb{E}_n \left[ \left( f_n^* - \min_{i=1, \dots, q} f(\mathbf{x}_i) \right)^+ \right], \quad (4)$$

where  $\mathbb{E}_n[\cdot] := \mathbb{E}[\cdot \mid \mathbf{x}^{(1:n)}, y^{(1:n)}]$  is the expectation taken with respect to the posterior distribution.

Ginsbourger et al. (2007) then proposes that we should choose to next evaluate the set of points that maximizes the multi-points expected improvement,

$$\operatorname{argmax}_{\mathbf{X} \in \mathbb{A}^q} q\text{-EI}(\mathbf{X}). \quad (5)$$

In the special case  $q = 1$ , which occurs when we are operating without parallelism, the multi-points expected improvement reduces to the expected improvement, as considered by Mockus (1989)

and Jones et al. (1998), and can be evaluated in closed-form, in terms of the normal pdf and cdf. The algorithm that chooses the next point to evaluate according to (5) is the EGO algorithm of Jones et al. (1998).

Ginsbourger et al. (2007) provided an analytical calculation of EI when  $q = 2$ , but in the same paper Ginsbourger commented that the general case of  $q$ -EI has complex expressions depending on  $q$ -dimensional Gaussian cumulative distribution functions, and computation of  $q$ -EI when  $q$  is large would have to rely on numerical multivariate integral approximation techniques, which is computationally expensive and makes solving (5) difficult. Ginsbourger (2009) writes "directly optimizing the  $q$ -EI becomes extremely expensive as  $q$  and  $d$  (the dimension of inputs) grow".

### 3 Algorithm

In this section we present a new algorithm for solving the inner optimization problem (5) of maximizing  $q$ -EI. This algorithm uses a novel estimator of the gradient of the  $q$ -EI presented in Section 3.2, used within a multistart stochastic gradient ascent framework as described in Section 3.3. We additionally generalize this technique from synchronous to asynchronous parallel optimization in Section 3.4. We begin by introducing some additional notation used to describe our algorithm.

#### 3.1 Notation

In this section we reformulate previously defined equations and define additional notation to better support construction of the gradient estimator.

We first write the posterior distribution on  $f(\mathbf{X})$  given by (2) in an alternative expression

$$f(\mathbf{X}) \stackrel{d}{=} \boldsymbol{\mu}(\mathbf{X}) + \mathbf{L}(\mathbf{X})\mathbf{Z}, \quad (6)$$

where  $\mathbf{L}(\mathbf{X})$  is the lower triangular matrix obtained from the Cholesky decomposition of  $\boldsymbol{\Sigma}^{(n)}$  in (2),  $\boldsymbol{\mu}(\mathbf{X})$  is the posterior mean (identical in meaning to  $\boldsymbol{\mu}^{(n)}$  in (2), but rewritten here to emphasize the dependence on  $\mathbf{X}$  and deemphasize the dependence on  $n$ ), and  $\mathbf{Z}$  is a  $q$ -dimensional standard normal random vector.

By substituting (6) into (4), we have

$$q\text{-EI}(\mathbf{X}) = \mathbb{E} \left[ \left( f_n^* - \min_{i=1,\dots,q} \mathbf{e}_i [\boldsymbol{\mu}(\mathbf{X}) + \mathbf{L}(\mathbf{X})\mathbf{Z}] \right)^+ \right], \quad (7)$$

where  $\mathbf{e}_i$  is a unit vector in direction  $i$  and the expectation is over  $\mathbf{Z}$ . To make (7) even more compact, define a new vector  $\mathbf{m}(\mathbf{X})$  and new matrix  $\mathbf{C}(\mathbf{X})$ ,

$$\begin{aligned} \mathbf{m}(\mathbf{X})_i &= \begin{cases} f_n^* - \boldsymbol{\mu}(\mathbf{X})_i & \text{if } i > 0, \\ 0 & \text{if } i = 0, \end{cases} \\ \mathbf{C}(\mathbf{X})_{ij} &= \begin{cases} -\mathbf{L}(\mathbf{X})_{ij} & \text{if } i > 0, \\ 0 & \text{if } i = 0, \end{cases} \end{aligned} \quad (8)$$

and (7) becomes

$$q\text{-EI}(\mathbf{X}) = \mathbb{E} \left[ \max_{i=0,\dots,q} \mathbf{e}_i [\mathbf{m}(\mathbf{X}) + \mathbf{C}(\mathbf{X})\mathbf{Z}] \right]. \quad (9)$$

### 3.2 Constructing the gradient estimator

We now construct our estimator of the gradient  $\nabla q\text{-}EI(\mathbf{X})$ . Let

$$h(\mathbf{X}, \mathbf{Z}) = \max_{i=0, \dots, q} \mathbf{e}_i [\mathbf{m}(\mathbf{X}) + \mathbf{C}(\mathbf{X})\mathbf{Z}]. \quad (10)$$

Then

$$\nabla q\text{-}EI(\mathbf{X}) = \nabla \mathbb{E}h(\mathbf{X}, \mathbf{Z}). \quad (11)$$

If gradient and expectation in (11) is interchangeable, the gradient would be

$$\nabla q\text{-}EI(\mathbf{X}) = \mathbb{E}\mathbf{g}(\mathbf{X}, \mathbf{Z}), \quad (12)$$

where

$$\mathbf{g}(\mathbf{X}, \mathbf{Z}) = \begin{cases} \nabla h(\mathbf{X}, \mathbf{Z}) & \text{if } \nabla h(\mathbf{X}, \mathbf{Z}) \text{ exists,} \\ 0 & \text{otherwise.} \end{cases} \quad (13)$$

$\mathbf{g}(\mathbf{X}, \mathbf{Z})$  can be computed using results on differentiation of the Cholesky decomposition from Smith (1995). We then propose to use  $\mathbf{g}(\mathbf{X}, \mathbf{Z})$  as our estimator of the gradient  $\nabla q\text{-}EI$ , and will discuss interchangeability of gradient and expectation, which implies unbiasedness of our gradient estimator, in Section 4.1. As will be discussed in Section 4.2, unbiasedness of the gradient estimator is one of the sufficient conditions for convergence of the stochastic gradient ascent algorithm proposed in Section 3.3.

### 3.3 Optimization of $q\text{-}EI$

Our stochastic gradient ascent algorithm begins with some initial point  $\mathbf{X}_0 \in \mathbb{A}^q$ , and generates a sequence  $\{\mathbf{X}_t : t = 1, 2, \dots\}$  using

$$\mathbf{X}_{t+1} = \prod_{\mathbb{A}^q} [\mathbf{X}_t + \epsilon_t \mathbf{G}(\mathbf{X}_t)], \quad (14)$$

where  $\prod_{\mathbb{A}^q}(\mathbf{X})$  denotes the closest point in  $\mathbb{A}^q$  to  $\mathbf{X}$ , and if the closest point is not unique, we use a closest point such that the function  $\prod_{\mathbb{A}^q}(\cdot)$  is measurable. Here, we restrict our search for the maximum of the  $q\text{-}EI$  to  $\mathbb{A}^q$  since the feasible set of the outer optimization problem is  $\mathbb{A}$ . One could also define a more restricted search region than  $\mathbb{A}^q$ , and revise (14) accordingly.  $\mathbf{G}(\mathbf{X}_t)$  is an estimate of the gradient of  $q\text{-}EI(\cdot)$  at  $\mathbf{X}_t$ . In our implementation of the algorithm, we use Monte Carlo simulation to estimate the gradient, averaging together  $M$  replicates of our stochastic gradient estimator,

$$\mathbf{G}(\mathbf{X}_t) = \frac{1}{M} \sum_{m=1}^M \mathbf{g}(\mathbf{X}_t, \mathbf{Z}_{t,m}), \quad (15)$$

where  $\{\mathbf{Z}_{t,m} : m=1, \dots, M\}$  are i.i.d. samples generated from the  $q$ -dimensional standard normal distribution, and  $\mathbf{g}(\mathbf{X}_t, \mathbf{Z}_{t,m})$  is defined in (13).  $\{\epsilon_t : t = 0, 1, \dots\}$  is a stochastic gradient stepsize sequence (Kushner and Yin 2003), typically chosen to be equal to  $\epsilon_t = \frac{a}{t^\gamma}$  for some scalar  $a$  and  $\gamma$ .

To find the global maximum of the  $q\text{-}EI$ , we use multiple restarts of the algorithm from a set of starting points, drawn from a Latin hypercube design (McKay et al. 2000), to find multiple stationary points, and then use simulation, written here with a fixed sample size  $N$  for simplicity, to identify the best stationary point found. We summarize the procedures in Algorithm 1.

One could replace this simple comparison based on  $N$  Monte Carlo samples with a more sophis-



---

**Algorithm 1** Optimization of  $q$ -EI

---

**Require:** number of starting points  $R$ ; stepsize constants  $a$  and  $\gamma$ ; number of steps for one run of gradient ascent  $T$ ; number of Monte Carlo samples for estimating the gradient  $M$ ; number of Monte Carlo samples for estimating  $q$ -EI  $N$ .

- 1: Draw  $R$  starting points from a Latin hypercube design in  $\mathbb{A}^q$ ,  $\mathbf{X}_{r,0}$  for  $r = 1, \dots, R$ .
  - 2: **for**  $r = 1$  to  $R$  **do**
  - 3:   **for**  $t = 0$  to  $T - 1$  **do**
  - 4:     Compute  $\mathbf{G}_t = \frac{1}{M} \sum_{m=1}^M \mathbf{g}(\mathbf{X}_{r,t}, \mathbf{Z}_{r,t,m})$  where  $\mathbf{Z}_{r,t,m}$  is a vector of  $q$  i.i.d. samples drawn from the standard normal distribution.
  - 5:     Update solution using stochastic gradient ascent  $\mathbf{X}_{r,t+1} = \mathbf{X}_{r,t} + \frac{a}{t^\gamma} \mathbf{G}_t$ .
  - 6:   **end for**
  - 7:   Estimate  $q$ -EI( $\mathbf{X}_{r,T}$ ) using Monte Carlo simulation with  $N$  i.i.d. samples, and store the estimate as  $\widehat{q\text{-EI}}_r$ .
  - 8: **end for**
  - 9: **return**  $\mathbf{X}_{r',T}$  where  $r' = \operatorname{argmax}_{r=1,\dots,R} \widehat{q\text{-EI}}_r$ .
- 

ticated ranking and selection algorithm discussed in Kim and Nelson (2007). We leave this out of Algorithm 1 for simplicity of description. One can also replace estimation of  $q$ -EI in Step 7 by a method using closed-form formula in Chevalier and Ginsbourger (2013).

In our implementation of this algorithm, we supply optional fallback logic. This fallback logic takes two additional parameters: a strictly positive real number  $\epsilon'$ , and an integer  $L$ . If  $\max_{i=1,\dots,R} \widehat{q\text{-EI}}_i \leq \epsilon'$ , so that multistart stochastic gradient ascent failed to find a point with estimated expected improvement better than  $\epsilon'$ , then we generate  $L$  additional solutions from a Latin Hypercube on  $\mathbb{A}^q$ , estimate the expected improvement at each of these using the same Monte Carlo approach as in Step 7, and select the one with the largest estimated expected improvement.

Analysis in Section 4.2 shows that under certain conditions, including unbiasedness of the gradient estimator, the stochastic gradient algorithm converges to a stationary point almost surely.

### 3.4 Asynchronous parallel optimization

(5) corresponds to synchronous parallel optimization, in which we wait for all  $q$  points from our previous batch to finish before searching for a new set of points. However, in some applications, we may wish to generate a new partial batch of points to evaluate next while  $p$  points are still being evaluated, before we have their values. This is common in expensive computer simulations, where simulations do not necessarily finish at the same time.

We can extend (5) to asynchronous parallel optimization: suppose parallelization allows a batch of  $q$  points to evaluate simultaneously; the first  $p$  points are still under evaluation, while the remaining  $q - p$  points have finished evaluation and the resources used to evaluate them are free to evaluate new points. We let  $\mathbf{X}' := (\mathbf{x}_1, \dots, \mathbf{x}_p)$  be the first  $p$  points still under evaluation, and let  $\mathbf{X} := (\mathbf{x}_{p+1}, \dots, \mathbf{x}_q)$  be the  $(q - p)$  points ready for new evaluations. Computation of  $q$ -EI for these  $q$  points remains the same as in (4), but we use an alternative notation,  $q\text{-EI}(\mathbf{X}', \mathbf{X})$ , to explicitly indicate that  $\mathbf{X}'$  are the points still being evaluated and  $\mathbf{X}$  are the new points to evaluate. Keeping  $\mathbf{X}'$  fixed, we optimize  $q$ -EI over  $\mathbf{X}$  by solving this alternative problem

$$\operatorname{argmax}_{\mathbf{X} \subset \mathbb{A}} q\text{-EI}(\mathbf{X}', \mathbf{X}). \quad (16)$$

As we did in the algorithm for synchronous parallel optimization in Section 3.3, we estimate the

gradient of the objective function with respect to  $\mathbf{X}$ , i.e.,  $\nabla_{\mathbf{X}} q\text{-}EI(\mathbf{X}', \mathbf{X})$ . The gradient estimator is essentially the same as in Section 3.2, except that we only take derivatives of  $h(\cdot, \cdot)$  with respect to  $\mathbf{X}$ . Then we proceed the same way as in Algorithm 1.

## 4 Theoretical analysis

In Section 3, when we constructed our gradient estimator and described the use of stochastic gradient ascent to optimize  $q\text{-}EI$ , we alluded to conditions under which this gradient estimator is unbiased and this algorithm converges to a stationary point of the  $q\text{-}EI$  surface. In this section, we describe these conditions and show unbiasedness and almost sure convergence to a stationary point of the  $q\text{-}EI$ .

### 4.1 Unbiasedness of the gradient estimator

We present a pair of lemmas and then our main theorem for unbiasedness of the gradient estimator. The lemmas will be used to support the proof of the theorem. Proofs of all results are available as supplemental material.

**Lemma 1.** *If a function  $\Psi : \mathbb{R} \mapsto \mathbb{R}$  is twice continuously differentiable over  $[-\epsilon, \epsilon]$ , where we define  $0/0 = 0$ , then for a sequence  $(\delta_\ell) \subseteq [-\epsilon, \epsilon]$ ,*

$$\sup_{\ell} \left| \frac{\Psi(\delta_\ell) - \Psi(0)}{\delta_\ell} \right| < \infty.$$

**Lemma 2.** *If  $\mathbf{m}(\mathbf{X})$  and  $\mathbf{C}(\mathbf{X})$  are differentiable in a neighborhood of  $\mathbf{X}$ , and there are no duplicated rows in  $\mathbf{C}(\mathbf{X})$ , then  $\nabla h(\mathbf{X}, \mathbf{Z})$  exists almost surely for any  $\mathbf{X}$ .*

**Theorem 1.** *If  $\mathbf{m}(\mathbf{X})$  and  $\mathbf{C}(\mathbf{X})$  are twice continuously differentiable in a neighborhood of  $\mathbf{X}$ , and  $\mathbf{C}(\mathbf{X})$  has no duplicate rows, then  $\nabla h(\mathbf{X}, \mathbf{Z})$  exists almost surely and*

$$\nabla \mathbb{E}h(\mathbf{X}, \mathbf{Z}) = \mathbb{E} \nabla h(\mathbf{X}, \mathbf{Z}).$$

The proof has a common approach for showing unbiasedness of an IPA estimator, by checking almost sure existence of the gradient estimator, and showing that the difference quotient is almost surely bounded by an integrable random variable (see (Glasserman 1991, Section 1.3.1)). Indeed, it is also possible to use previously published sufficient conditions for unbiasedness of IPA estimators within the proof of Theorem 1, such as (Glasserman 1991, Section 1.3.1), although this does not substantially simplify the proof, as most of the work in our proof is in verifying conditions required by this result.

Theorem 1 requires twice continuous differentiability of  $\mathbf{C}(\mathbf{X})$ , which seems difficult to verify because its analytic form is not obvious. However, following Smith (1995), which shows that  $m$ th-order differentiability of a symmetric and nonnegative definite matrix implies  $m$ th-order differentiability of the lower triangular matrix obtained from its Cholesky factorization,  $\mathbf{L}(\mathbf{X})$  has the same order of differentiability as  $\mathbf{\Sigma}^{(n)}$ . In addition, (8) shows that  $\mathbf{C}(\mathbf{X})$  has the same order of differentiability as  $\mathbf{\Sigma}^{(n)}$ , and therefore the order of differentiability for  $\mathbf{C}(\mathbf{X})$  can be verified through the order of differentiability of  $\mathbf{\Sigma}^{(n)}$ , which is determined by the covariance function  $k(\cdot, \cdot)$ .

## 4.2 Convergence analysis

In this section, we show almost sure convergence of our proposed stochastic gradient ascent algorithm. Although this algorithm assumed a search space of  $\mathbb{A}^q$ , we present the result where we perform the projection in (14) instead to a typically more flexible search space: a compact space  $H = \{\mathbf{X} : a_i(\mathbf{X}) \leq 0, i = 1, \dots, m\} \subseteq \mathbb{R}^{d \times q}$ , where  $a_i(\cdot)$  can be any real-valued constraint function. When  $\mathbb{A}$  is compact, and can be written as  $\mathbb{A} = \{\mathbf{x} : a'_i(\mathbf{x}) \leq 0, i = 1, \dots, m'\} \subseteq \mathbb{R}^d$ , where  $a'_i(\cdot)$  is any real-valued constraint function, then  $\mathbb{A}^q$  can be written in the form assumed for  $H$ . To do so, we write  $\mathbb{A}^q = \{\mathbf{X} : a_{i,j}(\mathbf{X}) \leq 0, i = 1, \dots, m', j = 1, \dots, q\}$ , where  $a_{i,j}(\mathbf{X}) = a'_i(\mathbf{x}_j)$  and  $\mathbf{x}_j$  is the  $j$ th point in  $\mathbf{X}$ .

The following theorem shows that under certain conditions, including the conditions from Theorem 1, the stochastic gradient ascent algorithm in Section 3.3 converges to a stationary point almost surely. The proof is available as supplemental material.

**Theorem 2.** *Suppose the following assumptions hold,*

1.  $a_i(\cdot), i = 1, \dots, p$  are twice continuously differentiable.
2.  $\epsilon_n \rightarrow 0$  for  $n \geq 0$  and  $\epsilon_n = 0$  for  $n < 0$ ;  $\sum_{n=1}^{\infty} \epsilon_n = \infty$  and  $\sum_{n=0}^{\infty} \epsilon_n^2 < \infty$ .
3.  $\forall \mathbf{X} \in H$ , under the definition (10),  $\mathbf{m}(\mathbf{X})$  and  $\mathbf{C}(\mathbf{X})$  are twice continuously differentiable and  $\mathbf{C}(\mathbf{X})$  does not have duplicate rows.

*Then the sequence  $\{\mathbf{X}_n : n = 0, 1, \dots\}$  generated by algorithm (14) converges to a stationary point on the  $q$ -EI surface almost surely.*

## 5 Numerical results

In this section, we present numerical experiments demonstrating the usefulness of MOE-qEI. In Section 5.1 we compare MOE-qEI against a previously proposed heuristic from the literature, which does not attempt to solve the inner optimization problem exactly, and demonstrate that MOE-qEI is able to find better solutions to the outer optimization problem using fewer measurements. In Section 5.2 we show that MOE-qEI provides a nearly linear speedup over single-threaded EGO. In Sections 5.3 and 5.4 we compare against two previously proposed methods for solving the inner optimization problem: in Section 5.3 we show that MOE-qEI solves the inner optimization problem more efficiently than using closed-form evaluations of  $q$ -EI developed by Chevalier and Ginsbourger (2013) when the dimension of the problem and  $q$  is large; in Section 5.4 we show that the computational time of  $\nabla q$ -EI for MOE-qEI scales better in  $q$  than the closed-form evaluation proposed by Marmin et al. (2015), and we argue that using cheap noisy evaluations in a stochastic gradient ascent framework to solve the inner optimization problem can be more efficient than using expensive closed-form evaluations in a gradient-based optimization framework. We performed all numerical experiments using the implementation of MOE-qEI available in the open source software package “MOE”. This software package is available at <http://github.com/Yelp/MOE>.

Although we use noise-free function evaluations, in numerical experiments, the covariance matrix  $K(\cdot, \cdot)$  in (2) may be ill conditioned. To resolve this problem, we adopt a standard trick from Gaussian process regression and Bayesian optimization (Rasmussen and Williams 2006, Section 3.4.3): we manually impose a small amount of noise  $\sim \mathcal{N}(0, \sigma^2)$  where  $\sigma^2 = 10^{-4}$  and use the noisy version of Gaussian Process regression, which is almost identical to (2), except that  $K(\mathbf{x}^{(1:n)}, \mathbf{x}^{(1:n)})$  is replaced by  $K(\mathbf{x}^{(1:n)}, \mathbf{x}^{(1:n)}) + \sigma^2 I_n$  where  $I_n$  is the identity matrix (Rasmussen and Williams 2006, Section 2.2).

## 5.1 Comparison against Constant Liar algorithm

Constant Liar is a heuristic  $q$ -EI algorithm proposed by Ginsbourger et al. (2007), which uses a greedy approach to iteratively construct a batch of  $q$  points. At each iteration of this greedy approach, the heuristic uses the sequential EGO algorithm to find a point that maximizes the expected improvement. However, since the posterior used by EGO depends on points chosen for the current batch from previous iterations that have not yet been evaluated, Constant Liar imposes a dummy response (a “liar”) at this point, and updates the Gaussian Process model with this “liar” value. The algorithm stops when  $q$  points are added, and reports the batch for function evaluation.

There are three variants of Constant Liar (CL), which use three different strategies for choosing the liar value: CL-min sets the liar value to the minimum response observed so far; CL-max sets it to the maximum response observed so far; and CL-mix is a hybrid of CL-min and CL-max, and uses the liar value of the method that gets the highest  $q$ -EI. Among the three methods, CL-mix was shown by Chevalier and Ginsbourger (2013) to have the best overall performance, and so we compare MOE-qEI against CL-mix.

We first let both MOE-qEI and CL-mix minimize the Branin function from Dixon and Szegö (1978). This is a two dimensional function and has a global minimum 0.3979. For both algorithms, we start with 15 observations in the region  $\{(x_1, x_2) : -15 \leq x_1 \leq 15, -15 \leq x_2 \leq 15\}$  using Latin hypercube sampling from McKay et al. (2000), then we construct a Gaussian Process model using the squared exponential kernel (Rasmussen and Williams 2006, Section 4.2), fit the characteristic length-scale of the kernel using the maximum likelihood approach described in (Rasmussen and Williams 2006, Section 5.4), and compute the posterior distribution as described in Section 2.2. Within each iteration, either algorithm (MOE-qEI or CL-mix) generates a batch of points for sampling, and the characteristic length-scale of the kernel as well as the posterior distribution of the Gaussian Process model is updated using new samples. The value of the best point  $f_n^*$  found so far at each iteration  $n$  is the performance metric in our comparison and tracks how fast an algorithm finds the global minimum. We ran both algorithms 200 times on the Branin function, and show the sample mean and 95% confidence interval for the expected value of  $f_n^*$  (averaging across the randomness in the initial stage of samples, and any residual randomness due to stochastic gradient ascent) vs. the iteration  $n$  in Figure 1a. The figure shows that MOE-qEI converges faster than CL-mix with equal or better solution quality in all three parallel settings,  $q = 2, 4, 8$ .

To show results over a large class of test problems functions, we compare the average performance of the two algorithms, where the test objective function  $f$  is drawn at random from a Gaussian Process. We produced 200 random test objective functions from a 3-dimensional Gaussian Process, where the characteristic length-scale along each dimension is set to 5 (this number is arbitrarily chosen). To make these test problems more realistic, we did not reveal the characteristic length-scales to the Gaussian process model used by both MOE-qEI and CL-mix, instead letting the model learn the parameters from sampled points as we did when minimizing the Branin function. For each random test problem, we began with 10 observations using Latin hypercube sampling in the region  $\{(x_1, x_2, x_3) : -20 \leq x_1 \leq 20, -20 \leq x_2 \leq 20, -20 \leq x_3 \leq 20\}$ , and ran both MOE-qEI and CL-mix. Figure 1b shows the sample mean and 95% confidence interval for the expected value of  $f_n^*$  vs. the iteration  $n$ . We see that MOE-qEI significantly outperforms CL-mix on both convergence speed and solution quality.

Together, these two experiments show that, by fully solving the inner optimization problem (5), MOE-qEI converges faster to a solution with better quality than the heuristic method CL-mix.

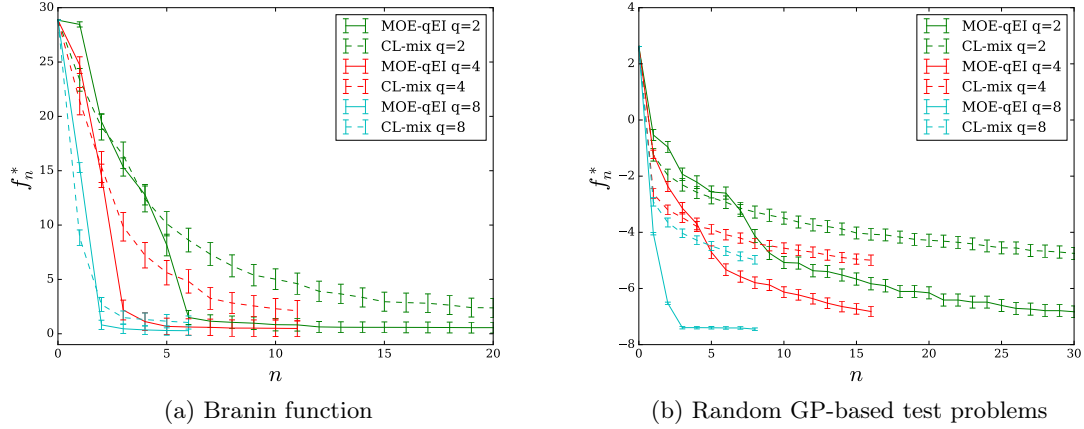


Figure 1: Expected solution value,  $f_n^*$ , vs. iteration  $n$ , in the outer optimization problem, under MOE-qEI and CL-mix, for three different levels of parallelism:  $q = 2, 4$ , and  $8$  threads. We are minimizing, and so smaller function values are better. MOE-qEI converges faster with better solution quality than the heuristic method CL-mix.

## 5.2 Comparison against EGO

Next, we compare MOE-qEI at different levels of parallelism against the fully sequential EGO algorithm, which is one-step optimal for sequential sampling. We use the same set of test functions and experiment setup as in Section 5.1. When  $q = 1$ , MOE-qEI is equivalent to EGO (where we use stochastic gradient ascent to maximize the expected improvement). Figure 2 shows the sample mean and 95% confidence interval for the expected value of  $f_n^*$  vs.  $n$  for EGO ( $q = 1$ ) and MOE-qEI ( $q > 1$ ) in minimizing Branin and random objective functions drawn from a Gaussian Process prior. Both experiments show that MOE-qEI obtains a substantial parallel speedup, and this speedup is nearly linear in  $q$ .

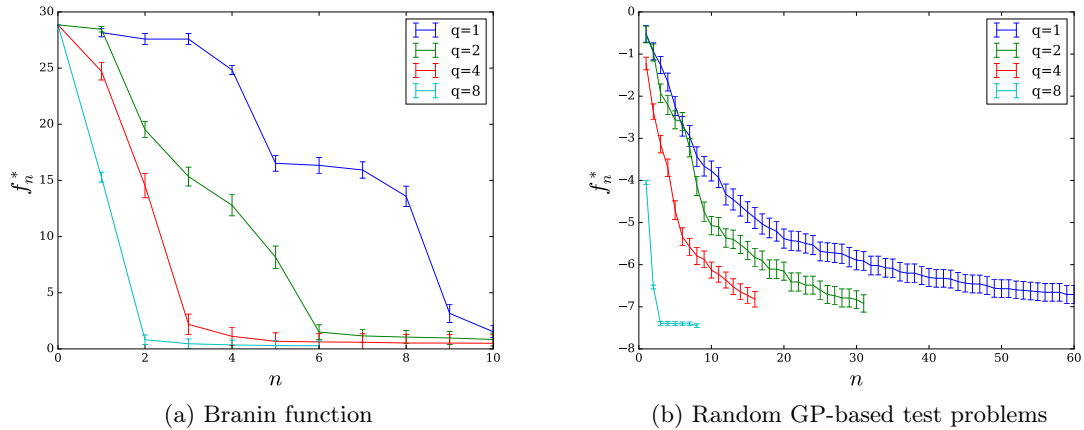


Figure 2: Expected solution value,  $f_n^*$  vs. iteration  $n$ , in the outer optimization problem, under EGO and MOE-qEI with different levels of parallelism  $q$ . MOE-qEI obtains a substantial speedup over EGO by evaluating in parallel, and the speedup is almost linear in  $q$ .

### 5.3 Comparison against closed-form evaluation of $q$ -EI

Chevalier and Ginsbourger (2013) provided a closed-form formula for  $q$ -EI and argued that it computes  $q$ -EI “very fast for reasonably low values of  $q$  (typically less than 10)”. While having a closed-form formula is appealing, calculating this formula becomes slow and numerically challenging even with moderately large  $q$ . This is because the formula requires  $q^2$  calls to the  $q - 1$  dimensional multivariate normal cdf, and computing multivariate normal cdfs with moderately large dimension is itself challenging, with state of the art methods relying on numerical integration or Monte Carlo sampling (see Genz (1992)).

While Chevalier and Ginsbourger (2013) did not propose using this closed-form formula to solve the inner optimization problem (5), one can adapt it to this purpose by using it within any derivative-free optimization method. We implemented this approach in MOE, where we use the L-BFGS Liu and Nocedal (1989) solver from SciPy (available at Jones et al. (2001)) as the derivative free optimization solver. We call this approach “Benchmark 1”.

We use numerical experiments to show that our Monte-Carlo based method solves the inner optimization more quickly and accurately than Benchmark 1. First we compare the speed with which Benchmark 1 and MOE- $q$ EI can solve the inner optimization. We fix  $q = 4$ , and let both methods start from randomly chosen points. We conducted experiments on both test problems drawn from a 2-dimensional Gaussian process and a 6-dimensional Gaussian process. We ran both inner optimization methods 800 times and show the sample mean and 95% confidence interval for the expected solution quality (the value of  $q$ -EI at the current iterate) vs. iteration (step) in Figure 3.

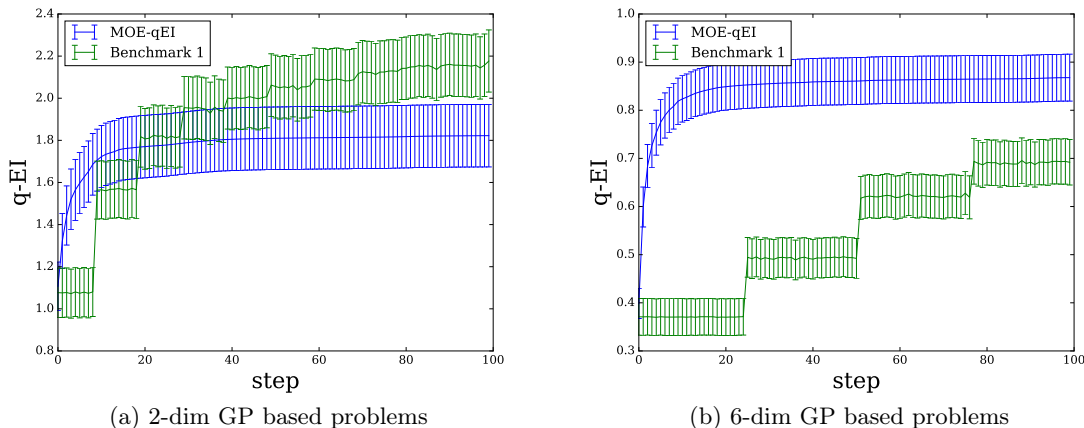


Figure 3: Expected solution quality vs. number of steps in the inner optimization problem under MOE- $q$ EI and Benchmark 1 (L-BFGS together with the closed form formula for  $q$ -EI from Chevalier and Ginsbourger (2013)). MOE- $q$ EI’s stochastic gradient ascent algorithm converges in fewer steps than L-BFGS in Benchmark 1, and each step is faster.

The figure shows that MOE- $q$ EI’s stochastic gradient ascent algorithm typically found points with better  $q$ -EI in fewer iterations than Benchmark 1’s L-BFGS thanks to the additional gradient information. Moreover, each iteration requires substantially less time under MOE- $q$ EI than under L-BFGS (approximately 1/10 of the time), because each step of the stochastic gradient algorithm requires only a single noisy evaluation of  $\nabla q$ -EI, while each step of Benchmark 1 requires multiple expensive closed-form evaluations of  $q$ -EI within a single step for gradient approximation and line search.

Second, we compare the quality of the solutions found to the inner optimization problem when using MOE-qEI and Benchmark 1. We use a similar experimental setup to the previous comparison, but fix the number of steps used by each method to 100. In this experimental comparison, we also include CL-mix, letting it use 100 steps in each of its greedy subproblems. We ran all three methods 250 times on 2-d problems and 575 times on 6-d problems.

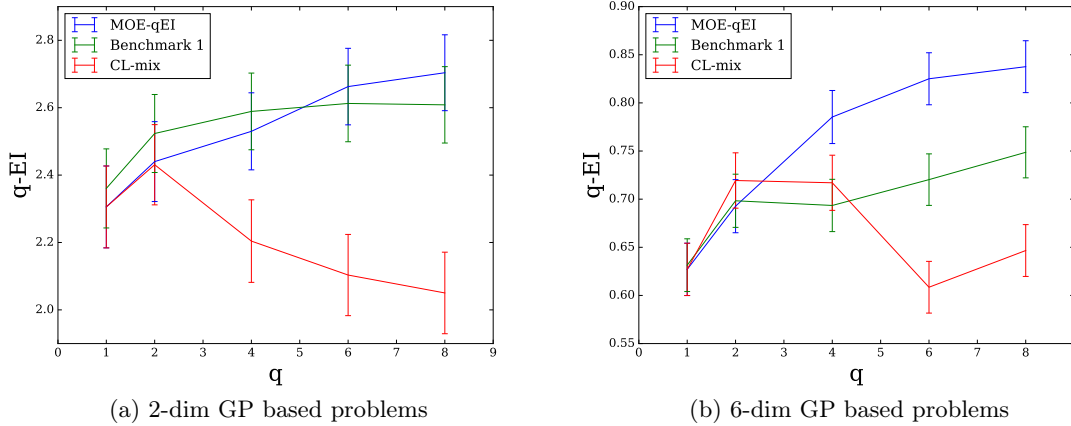


Figure 4: Expected solution quality after 100 steps in the inner optimization problem vs. level of parallelism  $q$ , under random test problems drawn from a Gaussian process prior in 2 and 6 dimensions. We compare MOE-qEI, Benchmark 1 (L-BFGS together with the closed form formula for  $q$ -EI from Chevalier and Ginsbourger (2013)), and CL-mix. As  $q$  and the dimension increase, MOE-qEI finds higher quality solutions.

Figure 4 shows sample means and 95% confidence intervals for the expected value of the  $q$ EI of the solution to the inner optimization problem calculated by each of these methods. The figure clearly indicates that CL-mix performs the worst, especially as  $q$  increases. This behavior is expected because CL-mix is a greedy heuristic, whose suboptimality increases as more greedy steps are added with increasing  $q$ . MOE-qEI and Benchmark 1 achieved similar solution quality on 2-d problems in the 100 iterations (though keeping in mind that iterations are more expensive under Benchmark 1), while on 6-d problems, MOE-qEI performs better when  $q \geq 4$ . This trend shows that as dimension and  $q$  increase and the inner optimization problem becomes more difficult, MOE-qEI tends to outperform Benchmark 1 by a larger margin.

#### 5.4 Comparisons against closed-form evaluation of $\nabla q$ -EI

A recently published book chapter Marmin et al. (2015), developed independently and in parallel to this work, proposed a method for computing  $\nabla q$ -EI using a closed-form formula, and then proposed using this inside a gradient-based optimization routine to solve (5). This closed-form formula faces computational challenges similar in spirit to those faced by the closed-form formula for  $q$ -EI proposed in Chevalier and Ginsbourger (2013), but even larger in magnitude. Indeed, this formula requires  $O(q^4)$  calls to multivariate normal cdfs with dimension between  $(q - 3)$  to  $q$ . Since computing high-dimensional multivariate normal cdfs is itself challenging, this closed-form evaluation becomes extremely expensive.

MOE-qEI’s Monte-Carlo based approach to evaluating  $\nabla q$ -EI offers three advantages over using the closed-form formula: first, numerical experiments below suggest that computation scales better

with  $q$ ; second, it can be easily parallelized, with significant speedups possible through parallel computing on graphical processing units (GPUs), as is implemented within the MOE library; third, by using a small number of replications to make each iteration fast, and by using it within a stochastic gradient ascent algorithm that averages noisy gradient information intelligently across iterations, we may more intelligently allocate effort across iterations, only spending substantial effort to estimate gradients accurately late in the process of finding a local maximum.

We first show that computation in MOE-qEI scales better with  $q$  through numerical experiments. We compare MOE-qEI with the implementation of closed-form gradient evaluation available in “DiceOptim” Ginsbourger et al. (2015), and call it “Benchmark 2”. We computed  $\nabla q\text{-EI}$  at 200 randomly chosen points from a 2-dimensional design space to obtain a 95% confidence interval for the average computation time. To make the gradient evaluation in MOE-qEI close to exact, we increased the number of Monte Carlo samples used in the gradient estimator to  $10^7$ , which ensures that the variance of each component of the gradient is on the order of  $10^{-10}$  or smaller for all  $q$  we have computed in the experiments. Figure 5 shows that computational time for Benchmark 2 increases quickly as  $q$  grows, but increases slowly for MOE-qEI’s Monte Carlo estimator, with this Monte Carlo estimator being faster when  $q \geq 4$ . This difference in performance arises because gradient estimation in MOE-qEI focuses Monte Carlo effort on calculating a single high-dimensional integral, while the closed-form formula decomposes this high-dimensional integral of interest into a collection of other high-dimensional integrals that are almost equally difficult to compute, and the size of this collection grows with  $q$ .

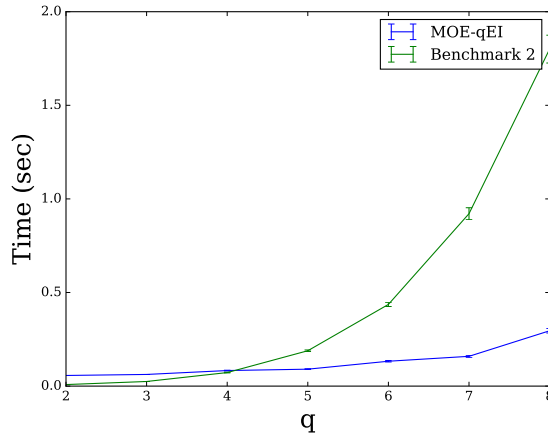


Figure 5: Average time to compute  $\nabla q\text{-EI}$  with high precision v.s.  $q$ , comparing the gradient-based estimator from MOE-qEI using a large number of samples ( $10^7$ ) with the closed-form formula from Marmin et al. (2015). The stochastic gradient estimator in MOE-qEI scales better in  $q$  and is faster when  $q \geq 4$ .

When we use this Monte Carlo estimator within MOE-qEI, we may obtain additional speed improvements by not using as many Monte Carlo samples. This is possible because stochastic gradient ascent is able to handle noisy gradient estimation. Using fewer Monte Carlo samples in each iteration increases efficiency by only putting effort toward estimating the gradient precisely when we are close to the stationary point, which stochastic gradient ascent performs automatically through its decreasing stepsize sequence. Indeed, we used  $10^6$  Monte Carlo samples for gradient estimation in all the other experiments we have conducted, which provides an order of magnitude speedup in each iteration, without dramatically increasing the number of iterations required.



## 6 Conclusions

We proposed an efficient method based on stochastic approximation for implementing a conceptual parallel Bayesian global optimization algorithm proposed by Ginsbourger et al. (2007). To accomplish this, we used infinitesimal perturbation analysis (IPA) to construct a stochastic gradient estimator and showed that this estimator is unbiased. We also provided convergence analysis of the stochastic gradient ascent algorithm with the constructed gradient estimator. Through numerical experiments, we demonstrate that our method outperforms the existing state-of-the-art approximation methods.

## References

- Amatriain, X. (2014). 10 lessons learned from building ml systems. <https://www.youtube.com/watch?v=WdzWPuazLA8>. Recording of presentation from MLconf 2014, Accessed: 2015-11-26.
- Boender, C. G. E. and Kan, A. R. (1987). Bayesian stopping rules for multistart global optimization methods. *Mathematical Programming*, 37(1):59–80.
- Brochu, E., Brochu, T., and de Freitas, N. (2010). A bayesian interactive optimization approach to procedural animation design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 103–112. Eurographics Association.
- Calvin, J. et al. (2005). One-dimensional global optimization for observations with noise. *Computers & Mathematics with Applications*, 50(1):157–169.
- Calvin, J. M. (1997). Average performance of a class of adaptive algorithms for global optimization. *The Annals of Applied Probability*, 7(3):711–730.
- Calvin, J. M. and Žilinskas, A. (2002). One-dimensional global optimization based on statistical models. In *Stochastic and Global Optimization*, pages 49–63. Springer.
- Chevalier, C. and Ginsbourger, D. (2013). Fast computation of the multi-points expected improvement with applications in batch selection. In *Learning and Intelligent Optimization*, pages 59–69. Springer.
- Clark, S. (2014). Introducing moe: Metric optimization engine; a new open source, machine learning service for optimal experiment design. <http://engineeringblog.yelp.com/2014/07/introducing-moe-metric-optimization-engine-a-new-open-source-machine-learning-service-for-optimal-ex.html>. Accessed: 2015-11-26.
- Clark, S. C., Liu, E., Frazier, P. I., Wang, J., Oktay, D., and Vedapant, N. (2014). Metrics optimization engine. <http://yelp.github.io/MOE/>. Accessed: 2016-02-28.
- Dennis, Jr, J. E. and Torczon, V. (1991). Direct search methods on parallel machines. *SIAM Journal on Optimization*, 1(4):448–474.
- Dixon, L. and Szegö, G. (1978). The global optimization problem: an introduction. *Towards global optimization*, 2:1–15.
- Frazier, P., Powell, W., and Dayanik, S. (2009). The knowledge-gradient policy for correlated normal beliefs. *INFORMS journal on Computing*, 21(4):599–613.
- Frazier, P. I. and Wang, J. (2016). Bayesian optimization for materials design. In *Information Science for Materials Discovery and Design*, pages 45–75. Springer.
- Frazier, P. I., Xie, J., and Chick, S. E. (2011). Value of information methods for pairwise sampling with correlations. In *Proceedings of the Winter Simulation Conference*, pages 3979–3991. Winter Simulation Conference.
- Genz, A. (1992). Numerical computation of multivariate normal probabilities. *Journal of computational and graphical statistics*, 1(2):141–149.
- Ginsbourger, D. (2009). Two advances in gaussian process-based prediction and optimization for computer experiments. In *MASCOT09 Meeting*.

- Ginsbourger, D., Le Riche, R., and Carraro, L. (2007). A multi-points criterion for deterministic parallel global optimization based on kriging. In *NCP07*.
- Ginsbourger, D., Picheny, V., Roustant, O., et al. (2015). Diceoptim: Kriging-based optimization for computer experiments. <https://cran.r-project.org/web/packages/DiceOptim/index.html>. Accessed: 2016-02-13.
- Glasserman, P. (1991). *Gradient estimation via perturbation analysis*. Springer Science & Business Media.
- Ho, Y.-C. (1987). Performance evaluation and perturbation analysis of discrete event dynamic systems. *IEEE Transactions on Automatic Control*, 32(7):563–572.
- Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.
- Howard, R. A. (1966). Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26.
- Huang, D., Allen, T. T., Notz, W. I., and Zeng, N. (2006). Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of global optimization*, 34(3):441–466.
- Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.
- Jones, E., Oliphant, T., Peterson, P., et al. (2001). SciPy: Open source scientific tools for Python. [Online; accessed 2014-12-01].
- Kennedy, J. (2010). Particle swarm optimization. In *Encyclopedia of Machine Learning*, pages 760–766. Springer.
- Kim, S.-H. and Nelson, B. L. (2007). Recent advances in ranking and selection. In *Proceedings of the 39th conference on Winter simulation: 40 years! The best is yet to come*, pages 162–172. IEEE Press.
- Kushner, H. and Yin, G. G. (2003). *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer Science & Business Media.
- Kushner, H. J. (1964). A new method of locating the maximum point of an arbitrary multippeak curve in the presence of noise. *Journal of Fluids Engineering*, 86(1):97–106.
- Liu, D. C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.
- Marmin, S., Chevalier, C., and Ginsbourger, D. (2015). Differentiating the multipoint expected improvement for optimal batch design. In *Machine Learning, Optimization, and Big Data*, pages 37–48. Springer.
- McKay, M. D., Beckman, R. J., and Conover, W. J. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61.
- Mockus, J. (1989). *The bayesian approach to local optimization*. Springer.
- Mockus, J., Tiesis, V., and Zilinskas, A. (1978). The application of bayesian methods for seeking the extremum. *Towards global optimization*, 2(117-129):2.
- Rasmussen, C. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. MIT Press.
- Royden, H. L. and Fitzpatrick, P. (1988). *Real analysis*, volume 198. Macmillan New York.
- Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science*, 4(4):409–423.
- Smith, S. P. (1995). Differentiation of the cholesky algorithm. *Journal of Computational and Graphical Statistics*, 4(2):134–147.
- Vazquez, E. and Bect, J. (2010). Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and inference*, 140(11):3088–3095.
- Villemonteix, J., Vazquez, E., and Walter, E. (2009). An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534.
- Xie, J., Frazier, P. I., and Chick, S. E. (2013). Bayesian optimization via simulation with pairwise sampling and correlated prior beliefs. *Operations Research*. to appear.

## Appendix: proofs of results in the main paper

*Proof of Lemma 1.* By Taylor's theorem,

$$\Psi(\delta_\ell) = \Psi(0) + \Psi'(0)\delta_\ell + \frac{\Psi''(r_\ell)}{2}\delta_\ell^2,$$

where  $r_\ell$  is between 0 and  $\delta_\ell$ . Then

$$\begin{aligned} \sup_\ell \left| \frac{\Psi(\delta_\ell) - \Psi(0)}{\delta_\ell} \right| &= \sup_\ell \left| \Psi'(0) + \frac{\Psi''(r_\ell)}{2}\delta_\ell \right|, \\ &\leq |\Psi'(0)| + \sup_\ell \left| \frac{\Psi''(r_\ell)}{2}\delta_\ell \right| \quad (\text{by the triangle inequality}). \end{aligned}$$

Since  $\delta_\ell \in [-\epsilon, \epsilon]$ , then  $r_\ell \in [-\epsilon, \epsilon]$ . We also have  $\Psi''(\cdot)$  continuous over  $[-\epsilon, \epsilon]$ , thus

$$\sup_\ell \left| \frac{\Psi''(r_\ell)}{2}\delta_\ell \right| \leq \epsilon \sup_\ell \left| \frac{\Psi''(r_\ell)}{2} \right| < \infty.$$

□

*Proof of Lemma 2.* Observe that

$$h(\mathbf{X}, \mathbf{Z}) = \mathbf{e}_{I^*} [\mathbf{m}(\mathbf{X}) + \mathbf{C}(\mathbf{X})\mathbf{Z}],$$

where  $I^* \in \operatorname{argmax}_{i=0, \dots, q} \mathbf{e}_i [\mathbf{m}(\mathbf{X}) + \mathbf{C}(\mathbf{X})\mathbf{Z}] := \mathcal{S}$ .

We claim that  $\nabla h(\mathbf{X}, \mathbf{Z})$  exists only if  $\mathbf{e}_I (\frac{\partial \mathbf{m}(\mathbf{X})}{\partial x_{ik}} + \frac{\partial \mathbf{C}(\mathbf{X})}{\partial x_{ik}} \mathbf{Z})$  are equal  $\forall I \in \mathcal{S}$ , and  $\forall i, k$  ( $k$  iterates over dimension). Therefore,

$$\begin{aligned} \mathbb{P}(\nabla h(\mathbf{X}, \mathbf{Z}) \text{ does not exist}) &\leq \mathbb{P}(|\mathcal{S}| \geq 2), \\ &\leq \frac{1}{2} \sum_{i \neq j} \mathbb{P}(\mathbf{e}_i [\mathbf{m}(\mathbf{X}) + \mathbf{C}(\mathbf{X})\mathbf{Z}] = \mathbf{e}_j [\mathbf{m}(\mathbf{X}) + \mathbf{C}(\mathbf{X})\mathbf{Z}]), \\ &= \frac{1}{2} \sum_{i \neq j} \mathbb{P}((\mathbf{C}(\mathbf{X})_{i\cdot} - \mathbf{C}(\mathbf{X})_{j\cdot}) \mathbf{Z} = m(\mathbf{X})_j - m(\mathbf{X})_i), \end{aligned}$$

where  $\mathbf{C}(\mathbf{X})_{i\cdot}$  is  $i$ th row of  $\mathbf{C}(\mathbf{X})$ .

Since  $\mathbf{C}(\mathbf{X})_{i\cdot} \neq \mathbf{C}(\mathbf{X})_{j\cdot}$ ,  $\{\mathbf{Z} : (\mathbf{C}(\mathbf{X})_{i\cdot} - \mathbf{C}(\mathbf{X})_{j\cdot}) \mathbf{Z} = m(\mathbf{X})_j - m(\mathbf{X})_i\}$  is subspace of  $\mathbb{R}^q$  with dimension smaller than  $q$ , thus

$$\mathbb{P}((\mathbf{C}(\mathbf{X})_{i\cdot} - \mathbf{C}(\mathbf{X})_{j\cdot}) \mathbf{Z} = m(\mathbf{X})_j - m(\mathbf{X})_i) = 0 \quad \forall i \neq j.$$

Hence

$$\mathbb{P}(\nabla h(\mathbf{X}, \mathbf{Z}) \text{ does not exist}) \leq 0 = 0.$$

□

*Proof of Theorem 1.* Without loss of generality, we only look at perturbation on  $k$ th dimension of  $m$ th point  $\mathbf{x}_m$  within  $\mathbf{X}$ , where  $m = 1, \dots, q$ . We use  $\mathbf{e}_{m,k}$  to denote direction of perturbation, and let  $\delta$  be the magnitude of perturbation, then the new set of points after perturbation is  $\mathbf{X} + \delta \mathbf{e}_{m,k}$ .

In a slight abuse of notation, we define

$$\begin{aligned} h(\delta, \mathbf{Z}) &:= h(\mathbf{X} + \delta \mathbf{e}_{m,k}, \mathbf{Z}), \\ \mathbf{m}(\delta) &:= \mathbf{m}(\mathbf{X} + \delta \mathbf{e}_{m,k}), \\ \mathbf{C}(\delta) &:= \mathbf{C}(\mathbf{X} + \delta \mathbf{e}_{m,k}), \\ I_{(\delta, \mathbf{Z})}^* &:= \min \left( \operatorname{argmax}_{i=0, \dots, q} e_i [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] \right). \end{aligned}$$

Then

$$h(\delta, \mathbf{Z}) = \max_{i=0, \dots, q} e_i [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] = e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}].$$

We also define

$$\begin{aligned} \Delta(\delta, \mathbf{Z}) &:= h(\delta, \mathbf{Z}) - h(0, \mathbf{Z}), \\ &= e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] - e_{I_{(0, \mathbf{Z})}^*} [\mathbf{m}(0) + \mathbf{C}(0) \mathbf{Z}]. \end{aligned}$$

Let  $\epsilon > 0$ . Consider a sequence  $(\delta_\ell) \subseteq [-\epsilon, \epsilon]$  and  $\delta_\ell \searrow 0$  as  $\ell \rightarrow \infty$ . We want to show  $\lim_{\ell \rightarrow \infty} \frac{\Delta(\delta_\ell, \mathbf{Z})}{\delta_\ell}$  exists almost surely, and

$$\lim_{\ell \rightarrow \infty} \mathbb{E} \left[ \frac{\Delta(\delta_\ell, \mathbf{Z})}{\delta_\ell} \right] = \mathbb{E} \left[ \lim_{\ell \rightarrow \infty} \frac{\Delta(\delta_\ell, \mathbf{Z})}{\delta_\ell} \right].$$

As a first step we show that  $\sup_\ell \left| \frac{\Delta(\delta_\ell, \mathbf{Z})}{\delta_\ell} \right|$  is bounded. For any  $\delta$  in the sequence  $(\delta_\ell)$ , we consider 2 cases,

- Case 1: If  $e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] \geq e_{I_{(0, \mathbf{Z})}^*} [\mathbf{m}(0) + \mathbf{C}(0) \mathbf{Z}]$ , then

$$\begin{aligned} |\Delta(\delta, \mathbf{Z})| &= e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] - e_{I_{(0, \mathbf{Z})}^*} [\mathbf{m}(0) + \mathbf{C}(0) \mathbf{Z}], \\ &\leq e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] - e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(0) + \mathbf{C}(0) \mathbf{Z}], \\ &= e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) - \mathbf{m}(0) + \mathbf{C}(\delta) \mathbf{Z} - \mathbf{C}(0) \mathbf{Z}], \\ &\leq \left| e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) - \mathbf{m}(0) + \mathbf{C}(\delta) \mathbf{Z} - \mathbf{C}(0) \mathbf{Z}] \right|. \end{aligned}$$

- Case 2: If  $e_{I_{(\delta, \mathbf{Z})}^*} [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] \leq e_{I_{(0, \mathbf{Z})}^*} [\mathbf{m}(0) + \mathbf{C}(0) \mathbf{Z}]$ , then

$$|\Delta(\delta, \mathbf{Z})| \leq \left| e_{I_{(0, \mathbf{Z})}^*} [\mathbf{m}(\delta) - \mathbf{m}(0) + \mathbf{C}(\delta) \mathbf{Z} - \mathbf{C}(0) \mathbf{Z}] \right|$$

by a similar argument.

In general, we have shown that,

$$|\Delta(\delta, \mathbf{Z})| \leq \sum_{i=0}^q |e_i [\mathbf{m}(\delta) - \mathbf{m}(0) + \mathbf{C}(\delta) \mathbf{Z} - \mathbf{C}(0) \mathbf{Z}]|,$$

so

$$\begin{aligned} \sup_{\ell} \left| \frac{\Delta(\delta_{\ell}, \mathbf{Z})}{\delta_{\ell}} \right| &\leq \sum_{i=0}^q \sup_{\ell} \left| \frac{\mathbf{e}_i [\mathbf{m}(\delta_{\ell}) - \mathbf{m}(0)]}{\delta_{\ell}} + \frac{\mathbf{e}_i [(C(\delta_{\ell}) - C(0)) \mathbf{Z}]}{\delta_{\ell}} \right|, \\ &\leq \sum_{i=0}^q \sup_{\ell} \left| \frac{m(\delta_{\ell})_i - m(0)_i}{\delta_{\ell}} \right| + \sup_{\ell} \left| \frac{C(\delta_{\ell})_{i\cdot} - C(0)_{i\cdot}}{\delta_{\ell}} \mathbf{Z} \right|. \end{aligned}$$

Define  $\mathbf{v}$  and  $\mathbf{V}$  such that

$$\begin{aligned} v_i &= \sup_{\ell} \left| \frac{m(\delta_{\ell})_i - m(0)_i}{\delta_{\ell}} \right|, \\ V_{ij} &= \sup_{\ell} \left| \frac{C(\delta_{\ell})_{ij} - C(0)_{ij}}{\delta_{\ell}} \right|. \end{aligned}$$

Then we have

$$\sup_{\ell} \left| \frac{\Delta(\delta_{\ell}, \mathbf{Z})}{\delta_{\ell}} \right| \leq \sum_{i=0}^q \left( v_i + \sum_{j=1}^q V_{ij} |Z_j| \right) =: W(\mathbf{Z}),$$

The conditions in Theorem 1 show that  $\mathbf{m}(\delta)$  and  $\mathbf{C}(\delta)$  are twice continuously differentiable over  $[-\epsilon, \epsilon]$ , and with  $(\delta_l) \subseteq [-\epsilon, \epsilon]$ , Lemma 1 assures that  $v_i < \infty$ ,  $V_{ij} < \infty \forall i, j$ . Therefore

$$\mathbb{E}[W(\mathbf{Z})] = \sum_{i=0}^q v_i + \sum_{i=0}^q \sum_{j=1}^q V_{ij} \mathbb{E}|Z_j| < \infty.$$

Then  $W(\mathbf{Z})$  is integrable. Also  $\lim_{\ell \rightarrow \infty} \frac{\Delta(\delta_{\ell}, \mathbf{Z})}{\delta_{\ell}} = \frac{\partial h(\mathbf{X}, \mathbf{Z})}{\partial X_{mk}}$  exists almost surely by Lemma 2, where  $X_{mk}$  denotes  $k$ th dimension of  $\mathbf{x}_m$  within  $\mathbf{X}$ . Then by Dominated Convergence Theorem (see Royden and Fitzpatrick (1988)),

$$\lim_{\ell \rightarrow \infty} \mathbb{E} \left[ \frac{\Delta(\delta_{\ell}, \mathbf{Z})}{\delta_{\ell}} \right] = \mathbb{E} \left[ \lim_{\ell \rightarrow \infty} \frac{\Delta(\delta_{\ell}, \mathbf{Z})}{\delta_{\ell}} \right]. \quad (17)$$

Since  $\delta_{\ell} \searrow 0$  as  $\ell \rightarrow \infty$ , (17) becomes

$$\frac{\partial \mathbb{E}h(\mathbf{X}, \mathbf{Z})}{\partial X_{mk}} = \mathbb{E} \frac{\partial h(\mathbf{X}, \mathbf{Z})}{\partial X_{mk}}, \quad (18)$$

Since (18) applies to any  $i, k$ ,  $\nabla h(\mathbf{X}, \mathbf{Z})$  exists almost surely, and

$$\nabla \mathbb{E}h(\mathbf{X}, \mathbf{Z}) = \mathbb{E} \nabla h(\mathbf{X}, \mathbf{Z}).$$

□

*Proof of Theorem 2.* We use convergence analysis result from (Kushner and Yin 2003, Section 5, Theorem 2.3) to prove our theorem, and we first state it using our notation and setting: the sequence  $\{\mathbf{X}_n\}$  produced by algorithm (14) converges to a stationary point almost surely if the following assumptions hold,

1.  $\epsilon_n \rightarrow 0$  for  $n \geq 0$  and  $\epsilon_n = 0$  for  $n < 0$ ;  $\sum_{n=1}^{\infty} \epsilon_n = \infty$
2.  $\sup_n \mathbb{E} |\mathbf{G}(\mathbf{X}_n)|^2 < \infty$

3. There are functions  $\lambda_n(\cdot)$  of  $\mathbf{X}$ , which are continuous uniformly in  $n$ , a continuous function  $\bar{\lambda}(\cdot)$  and random variables  $\beta_n$  such that

$$\mathbb{E}_n \mathbf{G}(\mathbf{X}_n) = \lambda_n(\mathbf{X}_n) + \beta_n,$$

and for each  $\mathbf{X} \in H$ ,

$$\lim_n \left| \sum_{i=n}^{m(t_n+t)} \epsilon_i [\lambda_i(\mathbf{X}) - \bar{\lambda}(\mathbf{X})] \right| = 0$$

for each  $t > 0$ , and  $\beta_n \rightarrow 0$  with probability one. The function  $m(t_n + \cdot)$  is defined in (Kushner and Yin 2003, Section 5.1).

4.  $\sum_i \epsilon_i^2 < \infty$ .
5. There exists a twice continuously differentiable real-valued function  $\phi(\cdot)$ , and  $\bar{\lambda}(\cdot) = -\phi_{\mathbf{X}}(\cdot)$ .

Now we prove that the 5 conditions stated above are indeed satisfied if the assumptions in Theorem 2 hold.

1. Condition 1 is satisfied by assumption 2 in Theorem 2. Construction of this sequence has been discussed in Section 3.3.
2. First we assume  $M = 1$ , then  $\mathbf{G}(\mathbf{X}_n) = \mathbf{g}(\mathbf{X}_n, \mathbf{Z})$ , and

$$\begin{aligned} \mathbb{E} |\mathbf{G}(\mathbf{X}_n)|^2 &= \mathbb{E} \sum_{m=1}^q \sum_{k=1}^d e_{m,k} \mathbf{G}(\mathbf{X}_n)^2, \\ &= \sum_{m=1}^q \sum_{k=1}^d \mathbb{E} \left( \frac{\partial h(\mathbf{X}, \mathbf{Z})}{\partial X_{mk}} \Big|_{\mathbf{X}=\mathbf{X}_n} \right)^2, \\ &= \sum_{m=1}^q \sum_{k=1}^d \mathbb{E} \left[ e_{I_{\mathbf{Z}}^*} \left( \frac{\partial \mathbf{m}(\mathbf{X})}{\partial X_{mk}} + \frac{\partial \mathbf{C}(\mathbf{X})}{\partial X_{mk}} \mathbf{Z} \right) \Big|_{\mathbf{X}=\mathbf{X}_n} \right]^2, \\ &\leq \sum_{m=1}^q \sum_{k=1}^d \mathbb{E} \sum_{i=0}^q \left[ e_i \left( \frac{\partial \mathbf{m}(\mathbf{X})}{\partial X_{mk}} + \frac{\partial \mathbf{C}(\mathbf{X})}{\partial X_{mk}} \mathbf{Z} \right) \Big|_{\mathbf{X}=\mathbf{X}_n} \right]^2, \\ &= \sum_{m=1}^q \sum_{k=1}^d \sum_{i=0}^q \mathbb{E} \left[ e_i \left( \frac{\partial \mathbf{m}(\mathbf{X})}{\partial X_{mk}} + \frac{\partial \mathbf{C}(\mathbf{X})}{\partial X_{mk}} \mathbf{Z} \right) \Big|_{\mathbf{X}=\mathbf{X}_n} \right]^2, \end{aligned}$$

where  $I_{\mathbf{Z}}^* = \arg \max_{i=0, \dots, q} e_i(\mathbf{m}(\mathbf{X}_n) + \mathbf{C}(\mathbf{X}_n)\mathbf{Z})$ . Since  $\mathbf{m}(\mathbf{X})$  and  $\mathbf{C}(\mathbf{X})$  are continuously differentiable for  $\forall \mathbf{X} \in H$  and  $H$  is compact,  $\left\| \frac{\partial \mathbf{m}(\mathbf{X})}{\partial X_{mk}} \right\|_{\infty} < \infty$  and  $\left\| \frac{\partial \mathbf{C}(\mathbf{X})}{\partial X_{mk}} \right\|_{\infty} < \infty$  for  $\forall \mathbf{X} \in H$ .

It is easy to see that  $\sup_n \mathbb{E} \left[ e_i \left( \frac{\partial \mathbf{m}(\mathbf{X})}{\partial X_{mk}} + \frac{\partial \mathbf{C}(\mathbf{X})}{\partial X_{mk}} \mathbf{Z} \right) \Big|_{\mathbf{X}=\mathbf{X}_n} \right]^2 < \infty$ , and then we can conclude that  $\sup_n \mathbb{E} |\mathbf{G}(\mathbf{X}_n)|^2 < \infty$ . If  $M > 1$ ,  $\mathbf{G}(\mathbf{X}_n)$  is an average of i.i.d. samples of  $\mathbf{G}(\mathbf{X}_n)$  for  $M = 1$  (call it  $\mathbf{G}^1(\mathbf{X}_n)$  for simplicity), then  $\mathbb{E} |\mathbf{G}(\mathbf{X}_n)|^2 = \frac{1}{M} \mathbb{E} |\mathbf{G}^1(\mathbf{X}_n)|^2$ . We have just showed that  $\sup_n \mathbb{E} |\mathbf{G}^1(\mathbf{X}_n)|^2$  is finite, and thus  $\sup_n \mathbb{E} |\mathbf{G}(\mathbf{X}_n)|^2$  is finite. Therefore, condition 2 is satisfied.

3. Since evaluation of  $\mathbf{G}(\mathbf{X}_n)$  in (15) does not depend on previous points in the sequence  $\{\mathbf{X}_n\}$ ,  $\mathbb{E}_n \mathbf{G}(\mathbf{X}_n) = \mathbb{E} \mathbf{G}(\mathbf{X}_n) = \mathbb{E} \mathbf{g}(\mathbf{X}_n, \mathbf{Z})$ . From the assumptions, we know Theorem 1 holds, then define a function  $\bar{\mathbf{g}}(\cdot)$  on  $H$ , such that  $\bar{\mathbf{g}}(\mathbf{X}) = \mathbb{E} \mathbf{g}(\mathbf{X}, \mathbf{Z}) = \nabla \mathbb{E} h(\mathbf{X}, \mathbf{Z})$ . We want to show  $\bar{\mathbf{g}}(\mathbf{X})$  is continuous on  $H$ .

Without loss of generality, we only look at perturbation on  $k$ th component of  $\mathbf{X}$ . Fix  $\mathbf{X}$ , let perturbation be  $\delta$ , and then only look at  $j$ th component of  $\bar{\mathbf{g}}(\cdot)$ . Here the index  $k$  refers to one dimension of a point within  $\mathbf{X}$ , and similar for  $j$ . We first define some notations for ease, let

$$\begin{aligned}\frac{\partial \mathbf{m}(\delta)}{\partial X_j} &:= \frac{\partial \mathbf{m}(\mathbf{X} + \delta \mathbf{e}_k)}{\partial X_j}, \\ \frac{\partial \mathbf{C}(\delta)}{\partial X_j} &:= \frac{\partial \mathbf{C}(\mathbf{X} + \delta \mathbf{e}_k)}{\partial X_j}, \\ I_{(\delta, \mathbf{Z})}^* &:= \min \left( \operatorname{argmax}_{i=0, \dots, q} \mathbf{e}_i [\mathbf{m}(\delta) + \mathbf{C}(\delta) \mathbf{Z}] \right).\end{aligned}$$

then

$$\begin{aligned}\Delta(\delta) &= \mathbf{e}_j [\bar{\mathbf{g}}(\mathbf{X} + \delta \mathbf{e}_k) - \bar{\mathbf{g}}(\mathbf{X})] \\ &= \mathbb{E} \mathbf{e}_j [\nabla h(\mathbf{X} + \delta \mathbf{e}_k, \mathbf{Z}) - \nabla h(\mathbf{X}, \mathbf{Z})] \\ &= \mathbb{E} \left[ \mathbf{e}_{I_{(\delta, \mathbf{Z})}^*} \left[ \frac{\partial \mathbf{m}(\delta)}{\partial X_j} + \frac{\partial \mathbf{C}(\delta)}{\partial X_j} \mathbf{Z} \right] - \mathbf{e}_{I_{(0, \mathbf{Z})}^*} \left[ \frac{\partial \mathbf{m}(0)}{\partial X_j} + \frac{\partial \mathbf{C}(0)}{\partial X_j} \mathbf{Z} \right] \right]\end{aligned}\tag{19}$$

Using similar argument as in the proof of Theorem 1, we can show

$$\begin{aligned}& \left| \mathbf{e}_{I_{(\delta, \mathbf{Z})}^*} \left[ \frac{\partial \mathbf{m}(\delta)}{\partial X_j} + \frac{\partial \mathbf{C}(\delta)}{\partial X_j} \mathbf{Z} \right] - \mathbf{e}_{I_{(0, \mathbf{Z})}^*} \left[ \frac{\partial \mathbf{m}(0)}{\partial X_j} + \frac{\partial \mathbf{C}(0)}{\partial X_j} \mathbf{Z} \right] \right| \\ & \leq \left| \mathbf{e}_{I_{(0, \mathbf{Z})}^*} \left[ \frac{\partial \mathbf{m}(\delta)}{\partial X_j} - \frac{\partial \mathbf{m}(0)}{\partial X_j} + \frac{\partial \mathbf{C}(\delta)}{\partial X_j} \mathbf{Z} - \frac{\partial \mathbf{C}(0)}{\partial X_j} \mathbf{Z} \right] \right| \\ & \leq \sum_{i=0}^q |\mathbf{e}_i [\boldsymbol{\delta}^m + \boldsymbol{\delta}^C \mathbf{Z}]|\end{aligned}\tag{20}$$

where  $\boldsymbol{\delta}^m = \frac{\partial \mathbf{m}(\delta)}{\partial X_j} - \frac{\partial \mathbf{m}(0)}{\partial X_j}$  and  $\boldsymbol{\delta}^C = \frac{\partial \mathbf{C}(\delta)}{\partial X_j} - \frac{\partial \mathbf{C}(0)}{\partial X_j}$ . Then

$$|\Delta(\delta)| \leq \sum_{i=0}^q \mathbb{E} |\mathbf{e}_i \boldsymbol{\delta}^m| + \mathbb{E} |\mathbf{e}_i [\boldsymbol{\delta}^C \mathbf{Z}]|.\tag{21}$$

Because  $\frac{\partial \mathbf{m}(\cdot)}{\partial X_j}$  and  $\frac{\partial \mathbf{C}(\cdot)}{\partial X_j}$  are continuous, then  $\forall \epsilon^m > 0$  and  $\forall \epsilon^C > 0$ , there exists  $\xi > 0$  such that  $\forall |\delta| < \xi$ ,  $|\delta_i^m| < \epsilon_i^m, i = 0, \dots, q$ ;  $|\delta_{ij}^C| < \epsilon_{ij}^C, i = 0, \dots, q, j = 1, \dots, q$ . Then for any  $\delta$

such that  $|\delta| < \xi$ ,

$$\begin{aligned}
\mathbb{E} |e_i [\delta^C \mathbf{Z}]| &= \mathbb{E} \left| \sum_{j=1}^q \delta_{ij}^C Z_j \right|, \\
&\leq \sum_{j=1}^q \mathbb{E} |\delta_{ij}^C Z_j|, \\
&\leq \sum_{j=1}^q \mathbb{E} |\delta_{ij}^C| |Z_j|, \\
&< \sum_{j=1}^q \epsilon_{ij}^C \mathbb{E} |Z_j|.
\end{aligned} \tag{22}$$

Then

$$\begin{aligned}
|\Delta(\delta)| &< \sum_{i=0}^q \epsilon_i^m + \sum_{i=0}^q \sum_{j=1}^q \epsilon_{ij}^C \mathbb{E} |Z_j|, \\
&= \sum_{i=0}^q \epsilon_i^m + \sqrt{\frac{2}{\pi}} \sum_{i=0}^q \sum_{j=1}^q \epsilon_{ij}^C.
\end{aligned} \tag{23}$$

Let  $\epsilon' = \sum_{i=0}^q \epsilon_i^m + \sqrt{\frac{2}{\pi}} \sum_{i=0}^q \sum_{j=1}^q \epsilon_{ij}^C$ , we conclude that  $\forall \epsilon' > 0$ , there exists  $\xi > 0$  such that  $\forall |\delta| < \xi, |\Delta(\delta)| < \epsilon'$ . Therefore  $\bar{\mathbf{g}}(\mathbf{X})$  is continuous on  $H$ . Let  $\lambda_n(\cdot) \equiv \bar{\lambda}(\cdot) \equiv \bar{\mathbf{g}}(\cdot)$ , and  $\beta_n = 0$ , then the first half of condition 3 is satisfied.

Since  $\lambda_n(\cdot) \equiv \bar{\lambda}(\cdot)$ , the second half of condition 3 is satisfied from the fact that the summand is 0.

4. Condition 4 is satisfied by assumption 2 in Theorem 2.
5. From the proof of condition 3, we know  $\bar{\lambda}(\cdot)$  is  $\bar{\mathbf{g}}(\cdot)$ , and thus  $\phi(\cdot)$  is just negative  $q$ -EI. Since  $\bar{\mathbf{g}}(\cdot)$  is continuously differentiable,  $\phi(\cdot)$  is twice continuously differentiable. Therefore, condition 5 is satisfied.

In conclusion, all conditions are satisfied and therefore  $\{\mathbf{X}_n\}$  converges to a stationary point almost surely.  $\square$